Odyssey 2018 The Speaker and Language Recognition Workshop
26-29 June 2018, Les Sables d'Olonne, France

End-to-end automatic speaker verification with evolving recurrent neural networks

Giacomo Valenti [1,2], *Adrien Daniel*[1], *Nicholas Evans* [2]

[1] NXP Semiconductors, Mougins, France
[2] EURECOM, Biot, France

giacomo.valenti@eurecom.fr, adrien.daniel@gmail.com, evans@eurecom.fr

Abstract

The state-of-the-art in automatic speaker verification (ASV) is undergoing a shift from a reliance on hand-crafted features and sequentially optimized toolchains towards end-to-end approaches. Many of the latest algorithms still rely on frame-blocking and stacked, hand-crafted features and fixed model topologies such as layered, deep neural networks. This paper reports a fundamentally different exploratory approach which operates on raw audio and which evolves both the weights and the topology of a neural network solution. The paper reports what is, to the authors' best knowledge, the first investigation of evolving recurrent neural networks for truly end-to-end ASV. The algorithm avoids a reliance upon hand-crafted features and fixed topologies and also learns to discard unreliable output samples. Resulting networks are of low complexity and memory footprint. The approach is thus well suited to embedded systems. With computational complexity making experimentation with standard datasets impracticable, the paper reports modest proof-of-concept experiments designed to evaluate potential. Results equivalent to those obtained using a traditional GMM baseline system and suggest that the proposed end-to-end approach merits further investigation; avenues for future research are described and have potential to deliver significant improvements in performance.

1. Introduction

Deep learning approaches to automatic speaker verification (ASV) have emerged in recent years and are now at the state of the art. Deep learning techniques have been explored in the context of: feature extraction [1, 2]; the learning of posteriors in a joint factor analysis framework [3]; the extraction of phonetically-aware frame posteriors as a replacement for the universal background model in an i-vector framework [4]; an alternative to i-vectors within a probabilistic linear discriminant analysis (PLDA) framework [5]; the estimation of hidden Markov model state posterior probabilities [6, 7]; PLDA back-end scoring [8].

A common characteristic to the above works is the use of deep learning techniques as a means of replacing specific, and often single elements of a more complex toolchain. While it has demonstrated the benefit of deep learning, this work may not be capitalizing on the true potential whereby deep learning is applied in a so-called end-to-end approach; current techniques, *e.g.* [9], still rely on hand-crafted features or pre-determined topologies whereas evolutive learning techniques can facilitate the application of neural networks to raw inputs and automatically optimize the topology according to the task at hand.

A growing number of attempts have been made to overcome the reliance on hand-crafted features. Most operate on spectral representations, *e.g.* [10, 11, 12]. While spectral representations stem from a linear transformation of the raw audio, and thus serve as an equivalent representation, these solutions still rely upon the frame-blocking of speech signals into fixed-length windows. While recurrent neural networks, *e.g.*, long short-term memory (LSTM) architectures, can exploit speech dynamics [13], frame-blocking remains a perhaps-questionable constraint. In this sense, the avoidance of frame-blocking may offer some potential to improve on current approaches. While the literature shows successful attempts to apply deep learning techniques to raw audio, *e.g.* [14, 15, 16, 17, 18], these works relate to speech and emotion recognition in addition to spoofing detection. To the best of the authors' knowledge, there is no equivalent work in ASV.

Also characteristic to almost all attempts to use deep learning for ASV is the use of pre-determined topologies, namely topologies chosen manually and empirically optimized. Most deep learning solutions involve a layered, hierarchical approach [19, 20] in which the number of layers, their connectivity (local or full), the number of units per layer and their activation function (linear, rectified, *etc.*) are all predetermined. Research from beyond the field of speech processing, *e.g.* [21], suggests that the use of pre-determined topologies may be a limitation. Studies related to image classification, for example, show that topologies can be learned automatically [22], albeit still in hierarchical fashion.

Solutions to these limitations are already available. A particular class of techniques known as topology and weight evolving artificial neural networks (TWEANNs) [23] use genetic learning algorithms to optimize not only the weights of a network but also its topology. This paradigm embraces the principles of natural evolution and selection and allows connections between any units, thereby completely avoiding the notion of hierarchical layers.

Motivated by recent work on evolving recurrent neural networks for audio processing and classification [24], by the increasing popularity of end-to-end learning [9, 19, 25, 26] and to address the limitations of hand-crafted features and pre-determined topologies, this paper reports what is believed to be the first application of TWEANNs to ASV. The approach operates directly on unprocessed, raw audio which is treated subsequently by evolving network structures before final classification, making for a *truly* end-to-end pipeline.

Accordingly, the objective of the work presented in this paper was not to outperform the existing state of the art, but

This work was completed while A. Daniel was still at NXP Semiconductors.

10.21437/Odyssey.2018-47

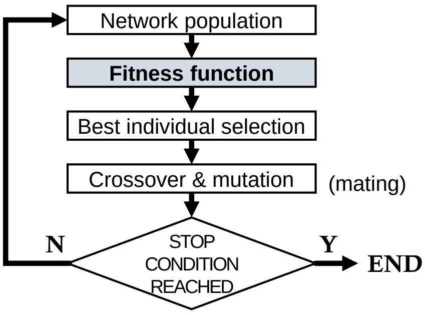


Figure 1: *An illustration of one iteration of evolution: the performance of each network in a population is assessed by the means of a fitness function and the* best *individuals are selected to form a new generation of networks.*

more specifically to investigate the longer term potential of the idea. The paper reports investigations with a particular form of TWEANN algorithms known as neuroevolution of augmenting topologies (NEAT) [27].

With the computational complexity of the algorithm far exceeding that of the established approaches to ASV (training only), experimentation with standard databases is currently impracticable. Implementation of the approach using efficient graphics processors is also far from being straightforward. In order to assess the potential of the idea, the paper reports modest proof-of-concept experiments designed to evaluate potential. The authors fully accept that the statistical significance afforded by such analysis is limited. With the algorithm representing something of a departure from current research directions and with results showing potential, the authors have elected to submit, admittedly early, the idea to the scrutiny of the scientific community.

Section 2 introduces the NEAT algorithm and its application to acoustic signals. Section 3 describes its adaptation and additional developments which are necessary such that the algorithm can be applied successfully to the ASV task. Experiments and results are described in Section 4. Conclusions are presented in Section 5.

2. Neuroevolution of augmenting topologies

The NEAT algorithm was introduced by Stanley and Miikkulainen in 2002 [27]. This section describes the main ideas behind the original work and then its previous application to audio classification problems.

2.1. Original algorithm

At a higher level, NEAT is a classical neuro-evolution algorithm which evolves a population of solutions (networks or individuals) according to an iterative process and a defined fitness function. Each iteration produces a new generation of solutions and the fitness function controls which among them serve as a basis to produce the next generation of solutions as shown in Fig 1.

At a lower level, however, NEAT is quite unique. One crucial aspect centers around the incremental evolution of structure. Even if the algorithm does not incorporate an explicit measure of complexity, networks tend to remain comparatively simple in structure compared to deep neural network solutions. Topologies are augmented iteratively in order to introduce diversity through the addition of new nodes and connections thereby following a *complexifying* principle. This is achieved through the usual biological analogies of mutation and crossover. These processes, in addition to that of weight mutation, are illustrated in Fig. 2.

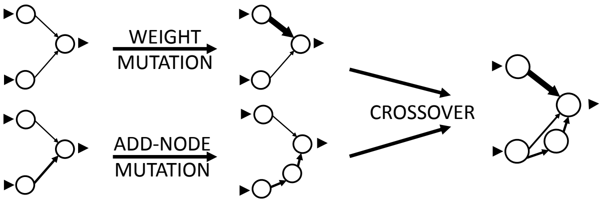


Figure 2: *Mutation of weight (here symbolized by connection thickness), node adding and crossover: the three forms of network evolution.*

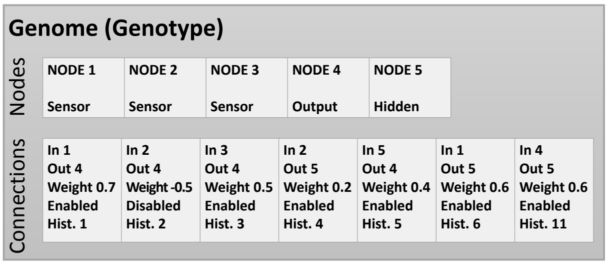
NEAT provides an elegant and efficient solution to a number of previously identified technical challenges such as the permutations problem [28]. These are addressed through the introduction of a genotype direct encoding scheme that features *historical markings* which track structural augmentations (see Fig. 3). Historical markings also serve a crucial purpose when performing crossover, as they provide a systematic means to align genes. Topology diversity is ensured by *speciation*, another biological analogy which protects structural innovation (*i.e.* by selecting the fittest networks within the same species).

With TWEANNs, weight and structural changes occur at random (within set boundaries) during evolution through mutation. The fitness function is an evaluation metric that aims to reward network changes which lead to improved performance. Hence the optimization process is not based on gradient descent and back-propagation, but instead consists in evaluating each network in the population according to the fitness function. To do so, training inputs are processed through the network exactly as they would be during inference, thereby yielding output values that serve to evaluate the network fitness. Then the best performing (i.e., the fittest) networks of the current population are selected to produce offspring for the next generation.

Since its conception, NEAT has been applied successfully to a multitude of tasks such as bipedal locomotion [29] and automated computer game playing [30]. NEAT continues to attract attention; shortly before the submission of this article, the authors became aware of recent, successful attempts to utilize NEAT for audio-related tasks such as audio effect creation [31] and sound event detection [32]. In view of the computational demands, however, in the former work NEAT was applied using a variety of classic spectral/cepstral frame-based features instead of raw audio, while the latter used wavelet representations.

2.2. Application to audio classification

Whereas the use of some form of frame-blocked spectral or filter-bank representation is characteristic to all previous work, Daniel [24] reported the first application of NEAT to audio classification which operates directly on time-domain inputs. NEAT is applied with networks constrained to a specific input/output setup and propagation scheme. As illustrated in Fig. 4, inputs consist of one or more streams of raw audio. Each stream is mapped to an input unit and is propagated through the network



Genome (Genotype)

Nodes

NODE 1	NODE 2	NODE 3	NODE 4	NODE 5
Sensor	Sensor	Sensor	Output	Hidden

Connections

In 1	In 2	In 3	In 2	In 5	In 1	In 4
Out 4	Out 4	Out 4	Out 5	Out 4	Out 5	Out 5
Weight 0.7	Weight -0.5	Weight 0.5	Weight 0.2	Weight 0.4	Weight 0.6	Weight 0.6
Enabled	Disabled	Enabled	Enabled	Enabled	Enabled	Enabled
Hist. 1	Hist. 2	Hist. 3	Hist. 4	Hist. 5	Hist. 6	Hist. 11

Figure 3: *A NEAT genotype is a direct and self-contained textual representation of a unique network, which contains (as in nature) more information than that which can be observed in the resulting structure. Figure reproduced with permission from [27].*

sample-by-sample with one activation step for every sample. An additional bias unit is set and held to unity. Network outputs consist of one or more *score* units whose outputs y_d are multiplied by the output of a binary *gate* unit y_r. Except for the score and gate output units, which have identity and binary step activations respectively, all units have rectified linear activation functions. The rate of the output (of any unit) is identical to that of the input, hence the networks perform one activation step per sample (see Fig. 4). In fact, the score output can be viewed as a new audio signal, the result of the network learning and applying to the input a transformation defined by the class to which the input belongs. The gate will thus evolve to discard *output* scores which are deemed to be unreliable, so that the network places emphasis on samples that are most helpful to discriminate between different audio classes. Alternatively, the gate can be replaced by a *reliability output* yielding a non-negative, non-binary weighting factor y_r. The operation of the gate/reliability output is similar in principle to that of attention mechanisms [33] which have been applied previously to speech recognition [34]. For each time sample i, the weighted mean over K samples of the product of y_d and y_r yields final weighted score y_w:

$$y_w[i] = \frac{\sum_{j=0}^{K-1} y_d[i-j] \times y_r[i-j]}{\sum_{j=0}^{K-1} y_r[i-j]} \tag{1}$$

The behavior of each network is assessed according to a generic squared-error-based fitness function F:

$$F(y_w, g) = 1/\left[1 + \sum_{i=0}^{N-1} (g[i] - y_w[i])^2\right] \tag{2}$$

which reflects the distance between N weighted scores y_w and a ground truth signal g of classification labels, *e.g.* 0 or 1, making for a supervised approach. Connections can be made freely between *any* pair of units. As a result, evolved networks may contain cyclical unit connections (e.g. units connected to themselves or to other units which influence their input). This classifies NEAT structures as *recurrent* neural networks.

3. End-to-end automatic speaker verification

This section reports the application of NEAT to conceive a truly end-to-end ASV system. In the work of [24], all networks are constrained to share the common setup and propagation scheme illustrated in Fig. 4: there is one input stream, one bias, one output stream and a binary gate. The process described in Section 2.2 is applied to generate networks which distinguish between a given target speaker and a set of background speakers. Each iteration of the algorithm corresponds to one independent evolutionary process applied in speaker-dependent fashion. This process will produce a population of increasingly discriminative, speaker-dependent networks.

The evolutionary process is driven according to a new fitness function which is introduced below. Also described in this section is a mini-batch procedure which was found to be beneficial to the evolution process. Specific training and testing procedures are also presented.

3.1. Fitness function

The fitness function in Eq. 2 does not necessarily reward separation between class distributions, but rather proximity to ground truth scores (*e.g.* 0 and 1). This behaviour becomes an evident problem when, after several generations, the two classes have only a minimal degree of overlap: a distance-based fitness function would reward a network that pushes the bulk of the distributions farther apart, without necessarily correcting previous classification errors; conversely, a network which fully separates classes but which produces noisier distributions would be attributed less reward than another network which produces pure Gaussian, but slightly overlapped distributions. An early search for an alternative, better suited to classification tasks such as ASV, investigated a fitness function based on the equal error rate (EER). The EER, though, only reflects the reliability of a classifier at a single operating point, *i.e.*, a fixed threshold. The area under the receiver operating characteristic curve (AUROC), in contrast, gives a measure of reliability which is independent from the operating point; it reflects the probability that the network will give a randomly chosen target sample a higher score than a randomly chosen impostor sample [36]. With notably better results, all work reported in this paper was performed by replacing Eq. 2 with an AUROC function calculated using the *trapezoid rule* [37].

3.2. Mini-batching

Inspired by a similar approach used in the *stochastic gradient descent* algorithm [38] to avoid over-fitting and convergence to local-optima, training is performed with a mini-batch process. The mini-batch process ensures that each generation of networks is trained using a different subset of data. This strategy promotes novelty during evolution since the training objective is changed every iteration. The same strategy also encourages generalization, namely networks which perform well across inter-session data. Finally, mini-batching also helps to reduce computational demands.

Each mini-batch consists of a fraction M_t of total target data and a fraction M_i of total impostor data. By way of example, with M_t=M_i=100%, every training iteration is performed using the *same* data; there is no mini-batching. With M_t=M_i=50%, training data is randomly shuffled and partitioned into two mini-batches. They are used in two subsequent iterations after which this process is repeated.

3.3. Training

The initial generation contains a population of 150 minimal, perceptron-like networks, each of which is configured according to the setup described in Section 2.2. The network weights

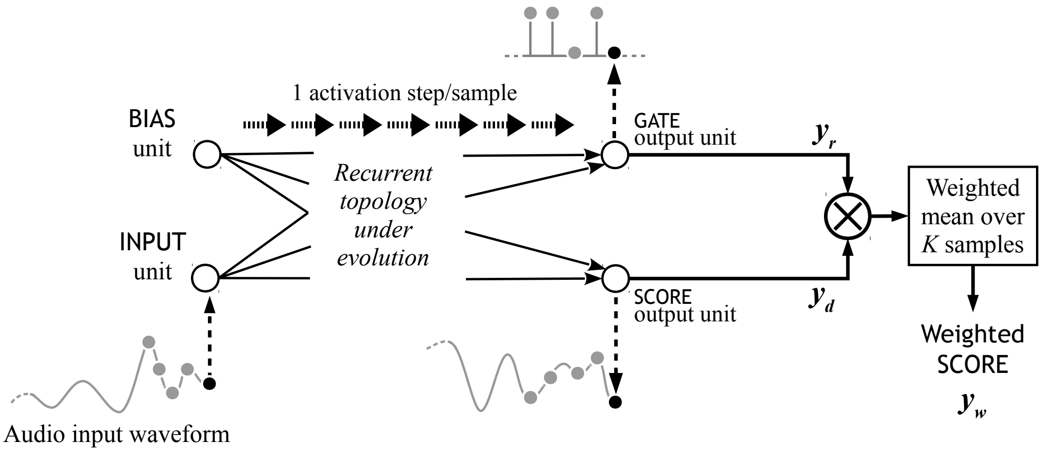


Figure 4: *End-to-end setup and propagation scheme for audio classification. There is one activation step per input sample; the output rate is the same as that of the input.*

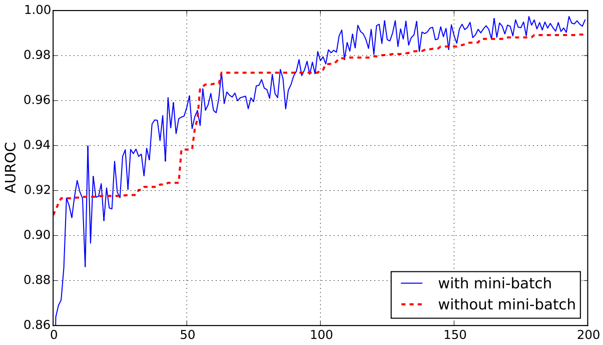


Figure 5: *An illustration of evolution measured in terms of fitness (AUROC) for the fittest network of each generation. The solid blue profile illustrates the AUROC with mini-batch training whereas the monotonic red profile shows the AUROC without mini-batch training.*

are randomly initialized and constrained within a $[-4, 4]$ range. Audio signals are normalized to within [-1, 1] in order to prevent saturation. This is more likely when using rectified linear activation functions as opposed to sigmoids, as in the original work. Rectified linear activation functions were found to be more efficient while giving similar performance. Every network in a given population is trained with the same mini-batch of data. Data containing either target or impostor speech is presented to each network in the form of non-contiguous segments of K samples. The system assigns to each segment a weighted mean score corresponding to $y_w[K-1]$ in Eq. 1. Networks are reset after the processing of each segment.

The fitness of each network is then determined according to the AUROC metric described in Section 3.1. The fittest networks of the population are then used to produce the next generation according to the procedure outlined in Section 2. The evolutionary process is applied iteratively until the fitness has converged.

Fig. 5 illustrates the evolution in fitness over 200 generations for an arbitrary target speaker. Each point on the graph corresponds to the population's fittest network for that generation. The solid blue profile illustrates evolution for the training procedure described above. Its non-monotonic nature is due to mini-batching; the data used at each iteration is different. The dashed red profile shows evolution with no mini-batching (M_t=M_i=100%); data used at each iteration is the same, hence the monotonic profile. While reducing processing time, mini-batching also results in faster learning.

3.4. Network selection for evaluation

Once training is complete, it is necessary to select and evaluate the single best network. First, the 10 best networks of each generation are identified according to the AUROC fitness function. Second, the performance of each of the 10 best networks from each generation is reassessed using the *full* training set. Since it gives a more intuitive interpretation of performance in a practical application, selection is performed using the application-neutral EER metric. The network which produces the lowest EER among the 10 is designated as the *generation champion*. Finally, the generation champion associated with the lowest EER is designated as the *grand champion*, and selected for evaluation. Evaluation is performed using an independent test set.

Aside from the fitness function and minor differences, this setup is also adopted in our own ongoing work in anti-spoofing [35].

4. Experiments

This section describes experiments which aim to test the potential of the end-to-end ASV system described in section 3.

4.1. Baseline system

The baseline system is a standard 64-component Gaussian mixture model (GMM) system [39]. Features are standard 19th order Mel-scaled frequency cepstral coefficients (MFCCs). These are appended with delta and double-delta parameters thereby giving features of 57 coefficients. Speaker models are derived from the maximum a posteriori adaptation of a universal background model (UBM). Scores are log-likelihood ratios given the speaker model and the UBM.

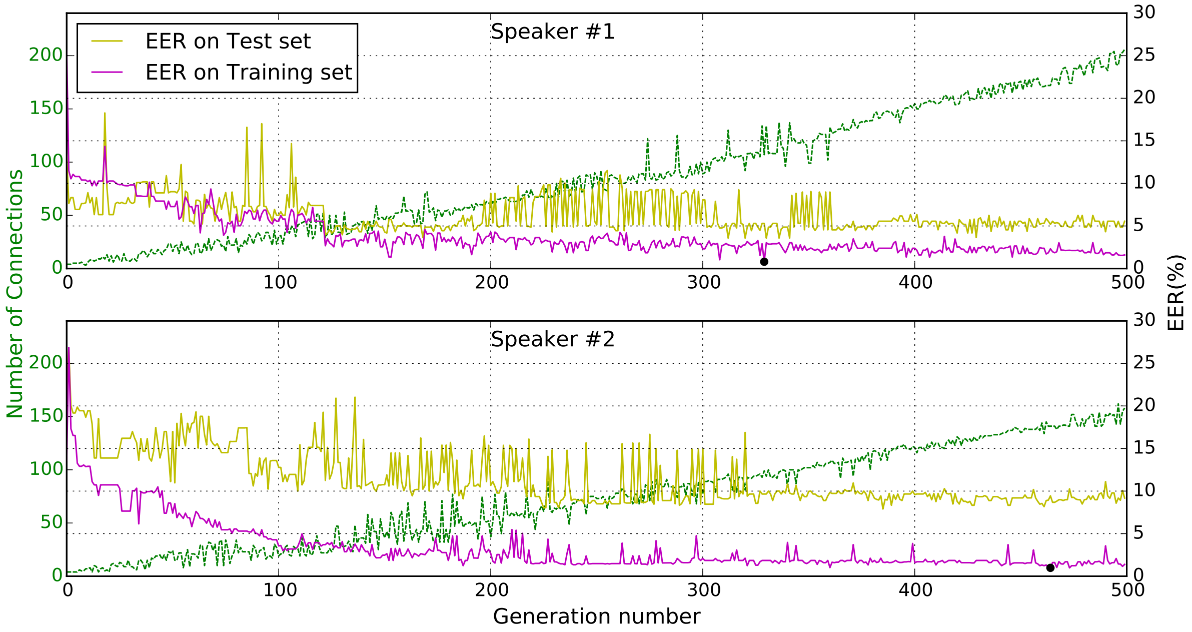


Figure 6: *Number of connections (green dashed profiles) and equal error rate (EER) of the first 500 generation champions for target speakers 1 (top) and 2 (bottom). EER profiles are shown for training data (magenta/dark) and testing data (yellow/light profiles). Black dots signify the grand champion, chosen according to the lowest-EER on the full training set.*

Table 1: *Results for the GMM and end-to-end systems in terms of EER for the training and test set for the two target speakers.*

	GMM Baseline		End-to-end system	
	Training	Test	Training	Test
Speaker #1	0%	9.52%	0.79%	5.30%
Speaker #2	0%	6.90%	0.98%	9.44%

4.2. NXP database and experimental protocols

Experimentation with standard NIST Speaker Recognition Evaluation (SRE) datasets [40], RSR [41] or RedDots [42] are currently impracticable on account of computational complexity. Being consistent with the objective to evaluate the potential of the algorithm, the paper reports a set of proof-of-concept experiments using a non-standard, proprietary database of speech signals collected from 10 male speakers. Text content consists of 10 of the 30 Harvard sentences which comprise the TIMIT database [43]. Each speaker provides approximately 5-6 minutes of speech which is recorded in 9 sessions over the course of one month. Recordings were collected in a quiet office with a laptop at a sampling rate of 16 kHz and 16-bit precision. Utterances were normalized by the active speech level estimated according to the ITU-T P.56 standard [44].

Among the 10 speakers, 2 are enrolled as targets. The training set consists of 6 of the 10 sentences uttered by the target speaker and the first 5 impostors. The test set consists of the other 4 sentences uttered by the target speaker and the remaining 3 impostors, thus achieving considerable phonetic separation between sets. Total target training data amounts to approximately 3.5 minutes of speech per speaker. Total impostor training data is in the order of 14 minutes duration.

For the end-to-end system, target data is partitioned into two mini-batches (M_t=50%). Since impostor data is more plentiful, it is partitioned into five mini-batches (M_i=20%) and used as background data for the baseline system. The average training recording is 3.25 seconds long. For the assessment and testing of both systems, one trial corresponds to one entire recording. Accordingly, K is set to $3.25\times16000 = 52000$ samples for training, and to each trial length at testing. Audio files used by the GMM system are preprocessed with silence removal. This step is not performed for the end-to-end system.

4.3. End-to-end system: augmentation and generalization

The training process took 17 and 13 hours for speaker 1 and 2, respectively, on an 8-core CPU running at 3.5 GHz. Several NEAT parameters influence the training time, e.g. without mini-batching, and with an otherwise identical setup, training takes several days.

Results obtained according to the evaluation procedure described in Section 3.4 are depicted in Fig. 6. Results are illustrated independently for the two target speakers and for 500 generations. The solid magenta (dark) profile in each plot shows the EER obtained by each generation champion assessed using the training data. EER profiles exhibit the expected evolution trend, namely a steady decrease from above 30% to less than 5% within 150 generations. The lowest EERs obtained by grand champion networks are 0.79% for speaker 1 (generation 329) and 0.98% for speaker 2 (generation 464) marked by black dots. Solid yellow (light) profiles show EERs for generation champions assessed on test data. As expected, performance on independent data is worse. Nonetheless, the selected grand champions are among the best performing networks on test data.

A summary of performance for both GMM and end-to-end systems is presented in Table 1. For the latter, results for both train and test datasets concern the grand champion network selected for each speaker. For the test set, grand champions yield EERs of 5.30% and 9.44%, whereas the GMM system delivers EERs of 9.52% and 6.90%. The gates of the grand champion networks prune an average of 46% of output data (in both speech and non-speech intervals) — the average for speaker 1 is 40% whereas that for speaker 2 is 53%. This percentage is consistently higher than than for the GMM system for which silence removal prunes an average of 35% of data. The effective behaviour of the gate was observed on a just few trials, depicting a periodic opening and closing as opposed to an energy- or amplitude-related activation. These findings show that (i) the performance of the end-to-end system is competitive with that of the GMM system and (ii) the two systems exploit data in a different way.

The upper green dashed profiles in Fig. 6 show the number of connections of each generation champion. As evolution proceeds, networks are steadily augmented with new nodes and connections. In general, network augmentations cause decreases in EERs for the training set, with 112 and 138 connections for speaker 1 and 2 grand champions, respectively. These networks are orders of magnitude less complex than usual, deep layered structures (*c.f.* $\sim 200k$ connections for the most compact model reported in [20]). Networks with such a reduced parameter space are inherently less prone to over-fitting since they do not have the capacity to learn a direct input-output correspondence.

5. Conclusions and future work

This paper reports an end-to-end approach to automatic speaker verification (ASV) based on the neuroevolution of augmenting topologies (NEAT) algorithm. In contrast to the existing state of the art, the proposed algorithm avoids the use of handcrafted features by processing raw audio and optimizes network weights and topologies in an entirely end-to-end fashion. Less complex topologies with a low memory footprint are well suited to embedded implementations. While reporting results for two speakers is not sufficient—and was neither intended—to provide a statistically reliable comparison between two systems, the proposed end-to-end approach is found to be at least competitive with a GMM baseline system.

These findings suggest that the end-to-end approach merits further attention and experimentation with larger, standard datasets. This work will require the reduction of computational efficiency; computational demands of the current algorithm make larger-scale experimentation impracticable.

In addition to the investigation of efficient implementations which exploit hardware acceleration, future work should consider non-binary gates for soft, rather than hard weighting of output score samples, and experimentation with longer duration training and testing. This work may well bring improvements in end-to-end system performance and/or expose application settings for which the proposed approach may excel.

6. References

[1] D. Yu and M. L. Seltzer, “Improved bottleneck features using pretrained deep neural networks,” in Interspeech, 2011, vol. 237, p. 240.

[2] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, “Deep speaker feature learning for text-independent speaker verification, in INTERSPEECH, 2017, pp. 1542-1546.

[3] S. Dey, S. Madikeri, M. Ferras, and P. Motlicek, “Deep neural network based posteriors for text-dependent speaker verification,” in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 5050-5054.

[4] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, “A novel scheme for speaker recognition using a phonetically-aware deep neural network,” in Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on, 2014, pp. 1695-1699.

[5] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, Deep neural network embeddings for text-independent speaker verification, in INTERSPEECH, 2017, pp. 999-1003.

[6] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 1, 2012, pp. 30-42.

[7] Y. Zhang, E. Chuangsuwanich, and J. R. Glass, “Extracting deep neural network bottleneck features using low-rank matrix factorization,” Acoustics, Speech and Signal Processing (ICASSP) IEEE International Conference on, 2014, pp. 185-189.

[8] H.-S. Lee, Y.-D. Lu, C.-C. Hsu, Y. Tsao, H.-M. Wang., and S.-K. Jeng, “Discriminative autoencorders for speaker verification”, in Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on, 2017.

[9] A. Miguel, J. Llombart, A. Ortega, and E. Lleida, “Tied hidden factors in neural networks for end-to-end speaker recognition, in INTERSPEECH, 2017, pp. 2819-2823.

[10] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, “Deep neural networks for small footprint text-dependent speaker verification,” in Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on, 2014, pp. 4052-4056.

[11] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, “Unsupervised feature learning for audio classification using convolutional deep belief networks,” in Advances in neural information processing systems, 2009, pp. 1096-1104.

[12] S.-X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, “End-to-End Attention based Text-Dependent Speaker Verification,” preprint arXiv:1701.00562, Jan. 2017.

[13] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition,” preprint arXiv:1402.1128, 2014.

[14] D. Palaz, R. Collobert, and M. M. Doss, “Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks,” eprint arXiv:1304.1018, 2013.

[15] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, “Learning the speech front-end with raw waveform CLDNNs,” in INTERSPEECH, 2015, pp. 1-5.

[16] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, “Acoustic modeling with deep neural networks using raw time signal for LVCSR,” in INTERSPEECH, 2014, pp. 890-894.

[17] G. Trigeorgis et al., “Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network,” in Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on, 2016, pp. 5200-5204.

[18] H. Muckenhirn, M. Magimai-Doss, and S. Marcel, “End-to-End Convolutional Neural Network-based Voice Presentation Attack Detection,” in IEEE IAPR International Joint Conference on Biometrics (IJCB), 2017.

[19] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, “End-to-end text-dependent speaker verification,” in Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on, 2016, pp. 5115-5119

[20] , “Locally-connected and convolutional neural networks for small footprint speaker recognition.,” in INTERSPEECH, 2015, pp. 1136-1140.

[21] Zhang, B.-T. and Muhlenbein, H., “Evolving optimal neural networks using genetic algorithms with Occams razor” in Complex Systems, no. 7, 1993, pp 199-220.

[22] B. Baker, O. Gupta, N. Naik, and R. Raskar, “Designing neural network architectures using reinforcement learning”, in International Conference on Learning Representations (ICLR), 2017.

[23] D. Dasgupta and D. McGregor, “Designing application-specific neural networks using the structured genetic algorithm”, In Proceedings of the International Conference on Combinations of Genetic Algorithms and Neural Networks, IEEE Press, Piscataway, New Jersey, 1992, pp 87-96.

[24] A. Daniel, “Evolving recurrent neural networks that process and classify raw audio in a streaming fashion”, in INTERSPEECH, 2017.

[25] A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks, in Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014, pp. 1764-1772.

[26] Y. LeCun, Y. Bengio and G. Hinton,“Deep learning”, in Nature, no. 521, 2015, pp 436-444.

[27] K. O. Stanley and R. Miikkulainen, “Evolving neural networks through augmenting topologies,” Evolutionary computation, vol. 10, no. 2, 2002, pp. 99-127.

[28] N. J. Radcliffe, “Genetic set recombination and its application to neural network topology optimisation,” Neural Computing and Applications, no. 1, 1993, pp. 67-90.

[29] B. Allen and Petros Faloutsos. “Complex networks of simple neurons for bipedal locomotion”, In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2009.

[30] M. Hausknecht, J. Lehman, R. Miikkulainen, and P. Stone, “A neuroevolution approach to general Atari game playing,” IEEE Transactions on Computational Intelligence and AI in Games, vol. 6, no. 4, 2014, pp. 355-366.

[31] I. Jordal,“Evolving artificial neural networks for cross-adaptive audio effects,” Masters Thesis, NTNU, 2017.

[32] C. Kroos and M. Plumbley,“Neuroevolution for sound event detection in real life audio: A pilot study,” Detection and Classification of Acoustic Scenes and Events (DCASE) Proceedings, 2017.

[33] C. Olah, and S. Carter, “Attention and augmented recurrent neural Networks”, in Distill, 2016.

[34] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, “Listen, attend and spell,” eprint arXiv:1508.01211v2, 2015.

[35] G. Valenti, H. Delgado, M. Todisco, N. Evans and L. Pilati, “An end-to-end spoofing countermeasure for automatic speaker verification using evolving recurrent neural networks”, Speaker Odyssey 2018.

[36] T. Fawcett, “An introduction to ROC analysis”, Pattern Recognition Letters, 27, 2006, pp. 861-874.

[37] J. A. C. Weideman, “Numerical integration of periodic functions: a few examples”, The American Mathematical Monthly, no. 109, 2002, pp. 21-36.

[38] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” in Proceedings of COMPSTAT2010, Springer, 2010, pp. 177-186.

[39] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker verification using adapted gaussian mixture models,” Digital Signal Processing, vol. 10, no. 13, Jan. 2000, pp. 19-41.

[40] S. O. Sadjadi et al., “The 2016 NIST Speaker Recognition Evaluation,” in INTERSPEECH, 2017, pp. 1353-1357.

[41] A. Larcher, K.-A. Lee, B. Ma, and H. Li, “RSR2015: Database for Text-Dependent Speaker Verification using Multiple Pass-Phrases.,” in INTERSPEECH, 2012.

[42] K. A. Lee et al., “The RedDots Data Collection for Speaker Recognition,” in Sixteenth Annual Conference of the International Speech Communication Association, 2015.

[43] Garofolo, J.S., Lamel, L.F., Fisher, W.M., Fiscus, J.G., Pallett, D.S., Dahlgren, N., Zue, V., “Timit acoustic-phonetic continuous speech corpus linguistic data consortium. Philadelphia, PA, 1993.

[44] International Telecommunication Union, “ITU-T P.56”, Series P: Terminals and Subjective and Objective Assessment Methods - Objective measurement of active speech level, 2011.